

# Comparative analysis of ARIMA and LSTM for predicting fluctuating time series data

Deddy Gunawan Taslim, I Made Murwantara

Department of Graduate Informatics, Faculty of Computer Science, Universitas Pelita Harapan, Jakarta, Indonesia

## Article Info

### Article history:

Received Feb 17, 2023

Revised Oct 18, 2023

Accepted Dec 6, 2023

### Keywords:

Autoregressive integrated moving average

Forecasting

Long short-term memory

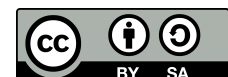
Time series

Volatile

## ABSTRACT

The investigation of time series data forecasting is a critical topic within the realms of economics and business. The autoregressive integrated moving average (ARIMA) model has been prevalently utilized, notwithstanding its limitations, which include the necessity for a substantial quantity of data points and the presumption of data linearity. However, with recent developments, the long short-term memory (LSTM) network has emerged as a promising alternative, potentially overcoming these limitations. The objective of this study is to determine an effective approach for managing time series data characterized by volatility and missing values. Evaluation was conducted using RMSE for accuracy assessment, and the execution time measured using the Python Timeit library. The findings indicates that in a dataset comprising 60 data points, the LSTM model (RMSE 0.037618) surpasses the ARIMA model (RMSE 0.062667) in terms of accuracy. However, this trend reverses in a larger dataset of 228 data points, where the ARIMA model demonstrates superior accuracy (RMSE 0.006949) compared to the LSTM model (RMSE 0.036025). In scenarios with missing data, the LSTM model consistently outperforms the ARIMA model, although the accuracy of both models diminishes with an increase in the number of missing values. The ARIMA model significantly outpaces the LSTM model.

*This is an open access article under the [CC BY-SA](#) license.*



## Corresponding Author:

I Made Murwantara

Department of Graduate Informatics, Faculty of Computer Science, Universitas Pelita Harapan

Jakarta, Indonesia

Email: made.murwantara@uph.edu

## 1. INTRODUCTION

Time series forecasting, which involves analyzing data sequences indexed in chronological order, holds significant relevance in fields such as economics, business, and finance. This domain has garnered considerable attention, especially in forecasting stock markets, supply-demand dynamics, and macroeconomic indicators based on historical trends and experiences [1]. Traditional statistical models, notably the autoregressive integrated moving average (ARIMA), have been widely employed for time series forecasting due to their relative ease of understanding and computational simplicity. However, ARIMA faces challenges, including the requirement of a certain volume of data for enhanced prediction accuracy [2], which may not always be practical in business contexts. Additionally, it operates under assumptions of data linearity and stationarity, limiting its effectiveness in highly irregular datasets [3]. Recent strides in computational capabilities and advancements in machine learning algorithms, particularly deep learning methods [4], have brought forward models like the

LSTM. These models show considerable promise in addressing the aforementioned challenges in time series forecasting [5]. Literature reviews reveal several comparative studies between ARIMA and LSTM in this context. Although deep learning-based methods are generally regarded as more adaptable to linear or non-linear data, few studies have scrutinized the effects of specific real-world data conditions. Concerning performance, there is a scarcity of research examining how these data conditions influence the run-time efficiency of ARIMA and LSTM models. Furthermore, the findings from existing research are inconclusive, with some studies favoring ARIMA and others indicating the superiority of LSTM in time series forecasting.

This research is driven by the quest for a versatile time series forecasting model capable of managing volatile data and maintaining satisfactory accuracy and computational efficiency. The aim is to compare ARIMA and LSTM models to identify a model meeting these criteria. The structure of this paper is as follows: section 2 delves into the literature review and foundational knowledge on the topic; section 3 describes the research methodology, including model design and experimentation; section 4 presents the results and analysis; and section 5 concludes the study, offering directions for future research.

## 2. BACKGROUND

### 2.1. Auto-regressive integrated moving average

The ARIMA, originally introduced by Box and Jenkins in 1970 [6], represents a comprehensive model that integrates the autoregressive and moving average (ARMA) processes to effectively analyze time series data [7]. The Box-Jenkins methodology, renowned for its iterative three-step process encompassing identification, estimation, and verification, has significantly contributed to time series data analysis. The evolution of computing technology has further enhanced the popularity and applicability of ARIMA, leading to its diversification into various forms such as univariate ARIMA models, transfer function models (also known as dynamic regression models), and multivariate (vector) ARIMA models.

ARIMA's application extends across a wide array of fields, including but not limited to business, economics, medicine, engineering, and other scientific domains [8], [9], as well as in predicting phenomena like computer network throughput [10]. A key advantage of ARIMA lies in its ability to leverage the temporal patterns inherent in time series data. It is also notably applied in scenarios involving non-stationary data. The acronym ARIMA encapsulates the fundamental components of the model: 'AR' denotes autoregression, a model assessing the dependencies of an observation on its previous values ( $p$ ); 'I' stands for integrated, a method used to induce stationarity in the time series by differencing observations at distinct intervals ( $d$ ); and 'MA' represents moving average, an approach that evaluates the dependency of observations on the residual errors derived from applying a simple moving average to lagged observations ( $q$ ).

### 2.2. Long short-term memory

LSTM networks, a specialized form of recurrent neural networks (RNNs), were introduced in 1997 [11] and are designed to emulate the learning processes of the human brain by retaining memory of previous inputs and discerning pivotal elements from non-essential ones [12]. LSTMs have found applications in a myriad of fields, including dynamic system modeling [13], image processing, speech recognition, sentiment analysis [14], and time series prediction [15]. An archetypal LSTM network, as depicted in Figure 1 [16], is composed of three distinct gates: the input gate, forget gate, and output gate. These gates operate on various elements such as the previous memory cell  $C_{t-1}$ , the current memory cell  $C_t$ , the previous output  $H_{t-1}$ , and the current output  $H_t$ , with  $\sigma$  symbolizing a sigmoid function and  $X_t$  representing the input at the current timestamp.

LSTMs are characterized by numerous hyperparameters [17], [18] that govern their training process and consequently affect their prediction accuracy or performance. In this study, a 'vanilla' LSTM configuration is employed, defined by predetermined hyperparameters including a set number of epochs at 25, units at 25, and the utilization of stochastic gradient descent (SGD) [19] as the optimizer. An epoch is a complete cycle through the training dataset [7], where one epoch indicates a single presentation of the dataset to the model, involving forward and backward data propagation and adjustment of network parameters' weights. A unit within an LSTM network comprises the LSTM cell itself along with the input, forget, and output gates [20]. SGD is a foundational method in neural network optimization. In SGD, gradients are computed for each network parameter across all dimensions, and parameter values are updated in the direction opposite to the gradient, proportional to the magnitude of the gradient [21].

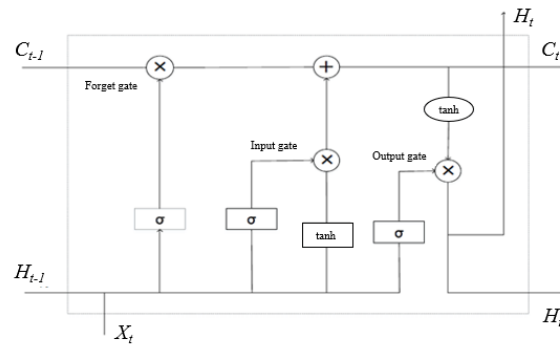


Figure 1. LSTM cell structure [16]

### 2.3. Method comparison and related studies

The ARIMA and LSTM models, both prevalent in time series forecasting, exhibit distinct characteristics and functionalities. As delineated in Table 1, these models present unique advantages and limitations. The ARIMA model is recognized for its simplicity, ease of interpretation, and straightforward application. However, its efficacy is notably diminished when applied to non-linear time series. In contrast, the LSTM model, although more intricate and challenging to interpret, demonstrates superior capability in handling non-linear time series data.

Table 1. ARIMA and LSTM comparison [22]

	ARIMA	LSTM
Underlying theory	Mathematics, statistics	Deep learning
Model complexity	Simple	Complex
Interpretability	Good	Poor
Non-linear processing ability	Poor	Good

#### 2.3.1. Stock price prediction

Gao's research explored stock price forecasting using the ARIMA model and various deep learning approaches, including Prophet, artificial neural networks (ANN), RNNs [23], and LSTM [24]. In a similar vein, Joosery and Deepa [25] conducted an analysis comparing ARIMA and LSTM. Their study also investigated the impact of data length on LSTM's predictive performance, observing that LSTM yields more favorable results when trained on a shorter time frame of three months' historical data. Conversely, the model's effectiveness diminishes with the use of longer time spans for training. Both studies employed mean square error (MSE) as the metric for evaluating model performance and arrived at a congruent conclusion that LSTM outperforms in stock price forecasting. Additionally, they noted indications suggesting that increasing the complexity of the neural network, such as implementing a sequence-to-sequence (Seq2Seq) model with attention, enhances prediction accuracy.

#### 2.3.2. Comparison of ARIMA and LSTM in time series forecasting

Namini *et al.* [7] conducted a comparative analysis of ARIMA and LSTM models using a rolling forecasting origin approach, specifically focusing on financial time series data. Their findings, as measured by The root mean square error (RMSE), revealed that models based on LSTM significantly surpassed those based on ARIMA, achieving an 85% reduction in error rates. Additionally, the study observed that varying the number of training iterations (epochs) did not yield any notable improvements in the model's performance.

#### 2.3.3. Research questions

The research questions addressed in this study are: firstly, does LSTM exhibit superior accuracy and run-time performance relative to ARIMA in datasets characterized by both short and long data point conditions? Secondly, does LSTM maintain its superiority in terms of accuracy and run-time performance over ARIMA in datasets that incorporate missing values?

### 3. METHOD

#### 3.1. Dataset

The research started with the base dataset of Indonesia's monthly consumer price index (CPI) from January 2003 to December 2021 sourced from the Bank of Indonesia (<https://www.bi.go.id/id/statistik/indikator/data-inflasi.aspx>). The base dataset was also intrinsically volatile as can be visually observed from sudden and spike of significant value changes as depicted in Figure 2. Further measure of the volatility in the dataset could also be observed from its coefficient of variation. Coefficient of variation is the ratio of the standard deviation to the mean, which measures the relative distribution of data points around the mean. Therefore, it can be concluded that the higher the coefficient of variation, the higher volatility was experienced in the dataset. As part of the first experiment on short versus long data points. The dataset was cut to appropriate length to simulate the short (60 data points), long (120 data points), and longer (228 data points). All further labelled as cpi\_60, cpi\_120, and cpi\_228 dataset correspondingly and visualised in Figure 3.



Figure 2. Indonesia's consumer price index

On the second experiment on missing values, the dataset of cpi\_60 and cpi\_120 were randomly imputed with zero values using the standard random.randint() function in Python. Number of zero values imputed was also at a varying rate to the total data points i.e., 10%, 25%, and 40% of the total data points. As expected, the missing values are also increasing the volatility in the data as measured in the coefficient of variance. The Table 2 and Figure 3 shows the comparison and visualisation of the datasets used in this research. All datasets were also split into train (in-sample) and test (out-of-sample) dataset on a 70:30 ratio. Therefore, a dataset with 60 data points will be split into 42 train and 18 test data points, and a dataset with 120 data points will be split into 96 train and 24 test data points and so on.

Table 2. Dataset comparison

Data set #	Dataset label	Number of data points	Missing values (%)	Coefficient of variance (%)
1.	cpi_60	60	0	47.54
2.	cpi_120	120	0	49.86
3.	cpi_228	228	0	58.7
4.	cpi60_mv0	60	0	47.54
5.	cpi60_mv10	60	10	59.92
6.	cpi60_mv25	60	25	75.39
7.	cpi60_mv40	60	40	89.66
8.	cpi120_mv0	120	0	49.86
9.	cpi120_mv10	120	10	62.75
10.	cpi120_mv25	120	25	77.66
11.	cpi120_mv40	120	40	88.43

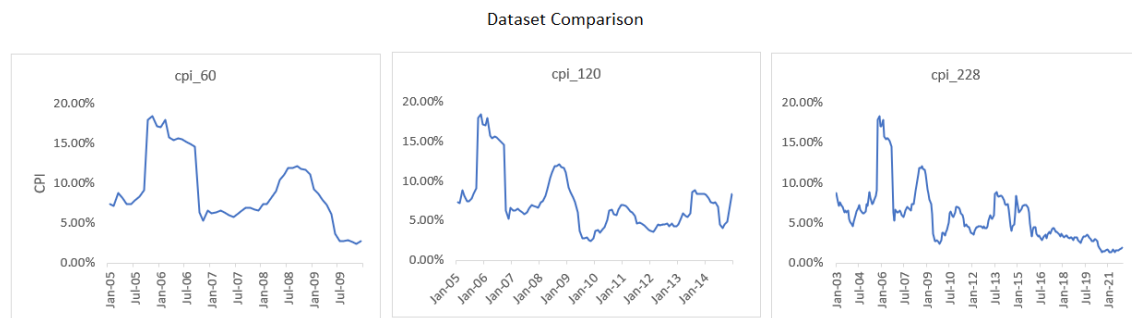


Figure 3. Comparison of volatile time series with different number of data points

### 3.2. Modeling

For this experiment, a basic ARIMA model was utilized with predefined parameters  $(p, d, q)$  set to  $(1, 1, 1)$ . The selection of these parameters was not aimed at optimizing the model's performance, but rather to establish a consistent control standard, maintained uniformly across the experiment. The measurement of the model's execution time was confined to the duration of the 'fit' command, deliberately excluding periods allocated to data preprocessing, model configuration, or evaluation procedures.

Similarly, a standard LSTM model was employed, with its hyper parameters set to a fixed number of epochs at 25, units at 25, and the use of SGD as the optimizer. Again, these hyper parameters were not adjusted for optimal results, but to serve as a stable baseline for the study. The timing of the LSTM model's operation was also recorded solely during the execution of the 'fit' command, explicitly excluding time spent on data preprocessing, configuring the model, or conducting evaluation tasks.

### 3.3. Performance metrics

#### 3.3.1. Model accuracy

For the assessment of the proposed models, the RMSE was employed as the evaluation metric. The choice to not utilize the mean absolute percentage error (MAPE) stems from its sensitivity and potential bias, particularly when the actual data includes values that are exceedingly small or proximate to zero. RMSE quantifies the square root of the average of the squared discrepancies between the predicted and the actual values.  $RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - x_i)^2}$  where  $n$  is the total number of observations,  $y_i$  is the actual value; and  $x_i$  is the predicted value.

#### 3.3.2. Run-time performance

Evaluating the run-time performance of each prediction algorithm is crucial for determining its applicability in real-time systems, as excessive prediction time can render an algorithm impractical for implementation. Within the confines of this research, all algorithms were implemented and executed multiple times on the Kaggle platform, equipped with 16 GB of memory, dual-core Xeon processors, and a 16 GB Tesla P100 GPU. The run-time assessment focused on the duration from the commencement of model fitting to the completion of generating prediction results, intentionally excluding the data preprocessing phase and the accuracy evaluation processes of the model. This run-time measurement was conducted using the standard Python 'timeit' library.

### 3.4. Experiments

#### 3.4.1. Experiment on short and long data points

In addressing the first research problem on whether LSTM are more superior in term of accuracy and run-time performance compared to ARIMA. Three datasets representing the short (60 data points), long (120 data points), and longer (228 data points), all further are labeled as cpi\_60, cpi\_120, and cpi\_228 were modeled with ARIMA and LSTM. All results were then compared and analyzed collectively.

#### 3.4.2. Experiment on missing values

For the second research problem, dataset with different length (60 and 120 data points) and varying rate of missing value (0%, 10%, 25%, and 40%) were represented by eight datasets and then modelled with ARIMA and LSTM. As part of control as well as simulating volatility in data, the missing value or imputation

with zero was intentionally not improved and all datasets were modelled as is. All results were then compared and analysed collectively.

## 4. RESULT AND DISCUSSION

### 4.1. Experiment on short and long data points

The outcomes of this study are detailed in Table 3 and depicted in Figure 4. Regarding model accuracy, ARIMA exhibited a consistent reduction in RMSE as the dataset expanded from 60 to 120 and then to 228 data points. In contrast, LSTM displayed a fluctuating RMSE trend, initially decreasing from 0.037618 to 0.024779 and then marginally increasing to 0.036025 with the dataset extension from 60 to 120 and 228 data points, respectively. With respect to modelling time, ARIMA showed minimal variations in run-time across different dataset sizes, whereas LSTM's modelling time escalated significantly from 9.9 to 15.2 and then to 35.9 seconds as the dataset grew.

Table 3. Results of experiment on short and long data points

Models	Dataset	RMSE errors	Modelling (seconds)
ARIMA	cpi_60	0.062667	0.100759
ARIMA	cpi_120	0.035722	0.115662
ARIMA	cpi_228	0.006949	0.122013
LSTM	cpi_60	0.037618	9.900116
LSTM	cpi_120	0.024779	15.19825
LSTM	cpi_228	0.036025	35.936281

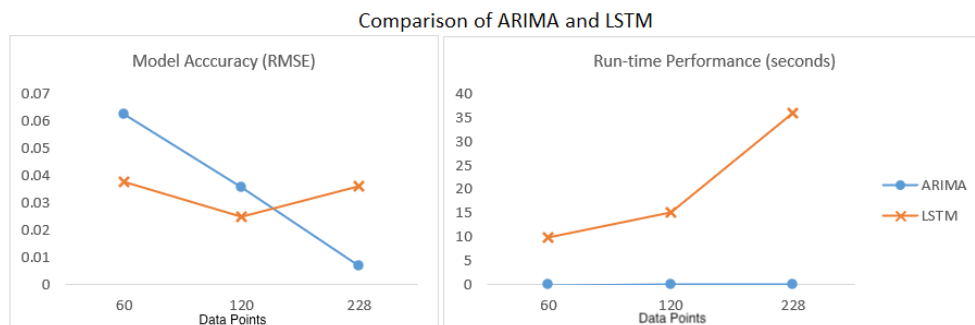


Figure 4. Results of experiment on short and long data points

The experiment demonstrated that ARIMA is less accurate with shorter datasets (60 data points) but shows substantial improvement as the data points increase. Specifically, ARIMA's accuracy improved by 43% and 89% when the dataset was extended to 120 and 228 data points, respectively, marking increases of 100% and 280% compared to the initial 60 data points dataset. This indicates ARIMA's higher accuracy with larger datasets.

LSTM, however, showed a 34% increase in accuracy when data points were expanded to 120 and only a 4% increase with 228 data points, compared to the base of 60 data points. This suggests that LSTM's accuracy does not necessarily improve with an increase in data points. Notably, all parameters in the modelling were consistently controlled across all runs.

LSTM outperformed ARIMA in terms of accuracy with shorter datasets (60 data points), but this trend reversed with longer datasets (228 data points). It is posited that ARIMA's moving average properties significantly impact its accuracy in short and volatile time series data, while LSTM, with its sequential properties and memory effect, maintains relative stability in accuracy regardless of dataset length. In terms of run-time performance, ARIMA consistently exhibited fast processing, around 0.1 seconds, regardless of the dataset size, attributed to its simpler calculations akin to linear functions. In contrast, LSTM's modelling time increased linearly with the dataset size, reflecting its complex RNN structure and the iterative loss minimization through backpropagation.

When comparing the two models, ARIMA was significantly faster (0.100759 seconds) than LSTM (9.900116 seconds) for shorter datasets (60 data points). This is in line with findings by Yamak *et al.* [26],

which also showed ARIMA's faster performance (0.0007 seconds) compared to LSTM (0.3267 seconds) in predicting Bitcoin price time series data. Although for shorter datasets, the modelling time may still be acceptable for non-real-time analysis.

The results are reported in Table 4 and visualised in Figure 5. In terms of modelling accuracy, ARIMA is generally affected by the number of missing values as it increases from 0% (no missing values) to 40%. Similar trend is also observed with LSTM being affected by the number of missing values in the dataset. In terms of modelling time, ARIMA has consistently seen negligible run-time differences regardless of the number of missing values in the dataset while LSTM's modelling time increased significantly from 9.9 to 11.8 seconds in dataset of 60 data points and from 15.2 to 52.1 seconds in dataset of 120 data points with varying increase of missing values.

Table 4. Results of experiment on missing values

Models	Dataset	RMSE errors	Modelling (seconds)
ARIMA	cpi60_mv0%	0.062667	0.100759
ARIMA	cpi60_mv10%	0.05572	0.098943
ARIMA	cpi60_mv25%	0.044377	0.094713
ARIMA	cpi60_mv40%	0.047862	0.128488
ARIMA	cpi120_mv0%	0.035722	0.115662
ARIMA	cpi120_mv10%	0.031919	0.092603
ARIMA	cpi120_mv25%	0.038224	0.080814
ARIMA	cpi120_mv40%	0.042859	0.12253
LSTM	cpi60_mv0%	0.037618	9.900116
LSTM	cpi60_mv10%	0.048626	11.442701
LSTM	cpi60_mv25%	0.048001	11.827139
LSTM	cpi60_mv40%	0.05087	11.865685
LSTM	cpi120_mv0%	0.024779	15.19825
LSTM	cpi120_mv10%	0.02504	37.510516
LSTM	cpi120_mv25%	0.025981	40.745405
LSTM	cpi120_mv40%	0.034824	52.135381

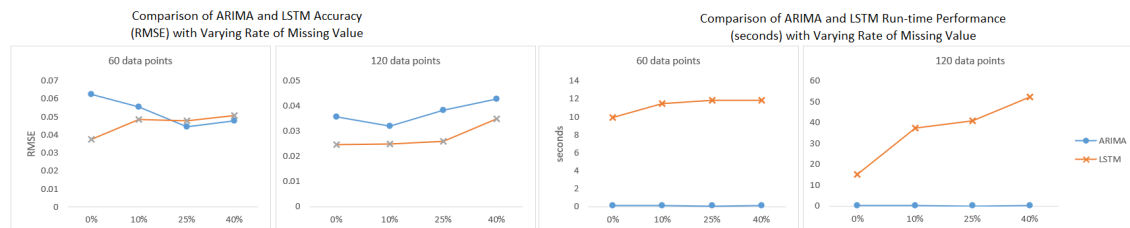


Figure 5. Accuracy and run-time performance results of experiment on missing values

By comparing the predictions of all eight datasets, it can be argued that ARIMA is more sensitively affected by the missing value (or zero values in this research). This is probably the most expected outcome of the arithmetic function of taking zero values into the moving average in the ARIMA. LSTM on the other hand also shows a certain degree of sensitivity against missing values, but also at a rather stable rate, indicating that LSTM is able to deal with the data anomaly (missing or zero values) relatively better than ARIMA. This is potentially important to avoid overfitting as the future data quality becomes better. Nevertheless, it is also important to note that data with missing values of more than 40% is probably not reliable to begin with and may need certain data quality improvement strategies. From this experiment, it was clearly shown that missing values were affecting the accuracy of the model of both ARIMA and LSTM. From the dataset of 120 data points, ARIMA experienced a worsening of 34% from the lowest to the highest RMSE measured, while LSTM experienced a worsening of 40% in RMSE when the rate of missing values was increased from 0% to 40%. In the lowest RSME measured from both models, LSTM outperformed by a 29% of being more accurate than the ARIMA model.

From the comparison as well, it can be seen that ARIMA is more adversely affected by missing values compared to LSTM from the dataset of 120 data points. It is possible that this is caused by the moving average function of ARIMA to take in zero values as valid data rather than outlier or potential data quality problem.



LSTM on the contrary has shown better results in the experiment with missing values than ARIMA. This is in line with the result of the first research problem where LSTM is found to be more stable in handling and predicting volatile data, possibly thanks to the neural network configuration and each round calculation adjusted the LSTM model to meet certain minimum loss function.

It is also important to note that the declining RMSE in the case of a dataset with 60 data points does not necessarily mean the accuracy is improving. Rather the increasing number of missing values which are then imputed with zero is causing the overall dataset to be smoothened as the CPI value ranged from 0.0241 to 0.1838 in the dataset `cpi_60`. This observation is further confirmed with the longer dataset of 120 data points and the increasing number of missing values is decreasing the accuracy of the modelling result significantly.

Similar to findings in the first research problem, ARIMA has shown to be much superior in terms of run-time performance achieving near 0.1 seconds regardless of the data condition (with or without missing values). This result is as expected of the nature of ARIMA's simpler mathematical model and calculation. LSTM on the other hand has shown significantly longer time to model especially when the number of data points and rate of missing values are both increased. LSTM model time took 15.2 to 52.1 seconds in a dataset of 120 data points, that is a 243% slower when there are no missing values in the data. While as discussed in previous section, LSTM evidently has a better ability of addressing the data anomaly (missing or zero values) than ARIMA, it is at the cost of run-time performance.

## 5. CONCLUSION

This research demonstrates that the ARIMA model can surpass the LSTM in accuracy when trained on an extensive dataset. Conversely, with shorter datasets, LSTM exhibits superior performance compared to ARIMA. When handling datasets with missing values, LSTM generally outperforms ARIMA, although it is noted that the accuracy of both models diminishes as the number of missing values increases. Regarding run-time efficiency, ARIMA consistently operates at a significantly faster rate, irrespective of the dataset size. The modelling time for LSTM, however, shows a proportional increase in relation to the volume of training data. This pattern in run-time performance is maintained even in scenarios involving datasets with missing values.

## ACKNOWLEDGEMENT

This project has been supported by the Graduate Informatics Program Grant No. 01679210019/2021/ FIK-MI/2023, Faculty of Computer Science, Universitas Pelita Harapan, Jakarta, Indonesia.

## REFERENCES




- [1] V. Kotu, B. Deshpande, M. Choi, and J. Lee, "Chapter 12 - time series forecasting," in *Data Science Second Edition*, pp. 395–445, 2019, doi: 10.1016/B978-0-12-814761-0.00012-5.
- [2] Z. Liu, Z. Zhu, J. Gao, and C. Xu "Forecast methods for time series data: A survey," *IEEE Access*, vol. 9, pp. 91896-91912, 2021, doi: 10.1109/ACCESS.2021.3091162.
- [3] A. Jalil and N. H. Rao, "Chapter 8 - time series analysis (stationarity, cointegration, and causality)," in *Environmental Kuznets Curve (EKC)*, pp. 85–99, 2019, doi: 10.1016/B978-0-12-816797-7.00008-4.
- [4] S. Özen, V. Atalay, and A. Yazici, "Comparison of predictive models for forecasting time-series data," in *ICBDR '19: Proceedings of the 3rd International Conference on Big Data Research*, 2019, pp. 172–176, doi: 10.1145/3372454.3372482.
- [5] B. A. Kalinggo and Zulkarnain, "Time series forecasting for non-stationary data: A case study of petrochemical product price," *APCORISE '20: Proceedings of the 3rd Asia Pacific Conference on Research in Industrial and Systems Engineering*, 2020, pp. 86–92, doi: 10.1145/3400934.3400952.
- [6] G. Box and G. Jenkins, "Time Series Analysis: Forecasting and Control," Holden-Day, 1976.
- [7] S. S. Namini, N. Tavakoli, and A. S. Namin, "A comparison of arima and lstm in forecasting time series," *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando, FL, USA, 2018, pp. 1394–1401, doi: 10.1109/ICMLA.2018.00227.
- [8] S. Noreen, S. Atique, V. Roy, and S. Bayne "Analysis and application of seasonal arima model in energy demand forecasting: A case study of small scale agricultural load," *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*, Dallas, TX, USA, 2019, pp. 521-524, doi: 10.1109/MWSCAS.2019.8885349.
- [9] L. M. Ang, K. P. Seng, G. K. Ijamaru, and A. M. Zungeru, "Auto regression integrated moving average (arima) to estimate temporal series of global solar radiation from measured data: Adrar region study," *ICIST '20: Proceedings of the 10th International Conference on Information Systems and Technologies*, 2020, pp. 1-4, doi: 10.1145/3447568.3448517.
- [10] D. Lee, D. Lee, M. Choi, and J. Lee, "Prediction of network throughput using arima," *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, 2020, pp. 1–5, doi: 10.1109/ICAIIIC48513.2020.9065083.
- [11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.






- [12] DiPietro, Robert, and G. D. Hager, "Deep learning: RNNs and LSTM," in *Handbook of Medical Image Computing and Computer Assisted Intervention*, Elsevier, pp. 503–519, 2020, doi: 10.1016/B978-0-12-816176-0.00026-0.
- [13] N. S. Malinovic, B. B. Predic, and M. Roganovic, "Multilayer long short-term memory (lstm) neural networks in time series analysis," in *2020 55th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, 2020, pp. 11–14, doi: 10.1109/ICEST49890.2020.9232710.
- [14] B. N. Saha and A. Senapati, "Long short term memory (lstm) based deep learning for sentiment analysis of english and spanish data," in *2020 International Conference on Computational Performance Evaluation (ComPE)*, pp. 442–446, 2020, doi: 10.1109/ComPE49325.2020.9200054.
- [15] W. Li, W. Y. Ng, T. Wang, M. Pelillo, and S. Kwong, "HELP: An LSTM-based approach to hyperparameter exploration in neural network learning," *Neurocomputing*, vol. 442, pp. 161–172, Jun. 2021, doi: 10.1016/j.neucom.2020.12.133.
- [16] Y. Wang and X. Mi, "A comparative study on demand forecast of car sharing users based on arima and lstm," *5th International Conference on Electromechanical Control Technology and Transportation (ICECTT)*, 2020, pp. 565–574, doi: 10.1109/ICECTT50890.2020.00130.
- [17] K. E. Hoque and H. Aljamaan, "Impact of Hyperparameter Tuning on Machine Learning Models in Stock Price Forecasting," in *IEEE Access*, vol. 9, pp. 163815–163830, 2021, doi: 10.1109/access.2021.3134138.
- [18] P. Rodriguez, "Improving the Stochastic Gradient Descent's Test Accuracy by Manipulating the Norm of its Gradient Approximation," *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023, pp. 1–5, doi: 10.1109/ICASSP49357.2023.10096624.
- [19] K. Agarwal, L. Dheekollu, G. Dhama, A. Arora, S. Asthana, and T. Bhowmik, "Deep learning based time series forecasting," *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Miami, FL, USA, 2020, pp. 859–864, doi: 10.1109/ICMLA51294.2020.00140.
- [20] S. Chakraborty, J. Banik, S. Addhya, and D. Chatterjee, "Study of dependency on number of lstm units for character based text generation models," *2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, 2020, pp. 1–5, doi: 10.1109/ICCSEA49143.2020.9132839.
- [21] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri, "diffgrad: An optimization method for convolutional neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4500–4511, 2020, doi: 10.1109/TNNLS.2019.2955777.
- [22] Z. Li, H. Zou, and B. Qi, "Application of arima and lstm in relative humidity prediction," in *2019 IEEE 19th International Conference on Communication Technology (ICCT)*, 2019, pp. 1544–1549, doi: 10.1109/ICCT46805.2019.8947142.
- [23] S. Mao and E. Sejdić, "A Review of Recurrent Neural Network-Based Methods in Computational Physiology," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 6983–7003, Oct. 2023, doi: 10.1109/TNNLS.2022.3145365.
- [24] Z. Gao, "Stock price prediction with arima and deep learning models," *2021 IEEE 6th International Conference on Big Data Analytics (ICBDA)*, Xiamen, China, 2021, pp. 61–68, doi: 10.1109/ICBDA51983.2021.9403037.
- [25] B. Joosery and G. Deepa, "Comparative analysis of time-series forecasting algorithms for stock price prediction," in *AISS '19: Proceedings of the 1st International Conference on Advanced Information Science and System*, 2019, no. 33, pp. 1–6, doi: 10.1145/3373477.3373699.
- [26] P. T. Yamak, L. Yujian, and P. K. Gadosey, "A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting," in *ACAI '19: Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*, 2019, pp. 49–55, doi: 10.1145/3377713.3377722.

## BIOGRAPHIES OF AUTHORS



**Deddy Gunawan Taslim**    is a graduate candidate in the Informatics Graduate Program. His areas of interest include operational research, data science, and artificial intelligence. He holds a Bachelor's degree in Computing from the President University in Indonesia and Master of Informatics from Department of Graduate Informatics Universitas Pelita Harapan. He can be contacted at email: 01679210019@student.uph.edu.



**I Made Murwantara**    is Department Chair of Graduate Informatics, Faculty of Computer Science, Universitas Pelita Harapan, Indonesia. His areas of research interest include software engineering for and in cloud computing, data science, and search-based software engineering. He earned a Ph.D. degree in Computer Science from the University of Birmingham, UK. He can be contacted at email: made.murwantara@uph.edu.